BAB II

Tinjauan Pustaka dan Dasar Teori

2.1 Tinjauan Pustaka

Tinjauan pustaka yang digunakan sebagai bahan acuan untuk melakukan studi komparasi ini diantaranya penelitian yang dilakukan oleh Verdi Yasin, Dian Agustina, dan Adiansyah dalam jurnal penelitiannya yang berjudul "Rancang Bangun Aplikasi Sistem *Audio Watermaking* dengan Metode *Phase Coding*". Dalam penelitianya metode *Phase Coding* menghasilkan kualitas suara yang baik karna memiliki nilai PSNR diatas 30, dan juga metode ini memiliki ketahanan terhadap serangan atau pengujian *Resampling,* dan penambahan derau, namun tidak tahan dengan serangan *croping dan time streching*.

Selain itu tinjauan pustaka dari penelitian Muhammad Soleh (2010) dalam skripsinya yang berjudul "Analisis dan Implementasi *Watermaking* dengan Algoritma *AES* untuk Pemberian Data Hak Cipta pada File *Audio*". Dalam penelitianya menjelaskan adanya pengaruh terhadap kualitas file *audio* dari file yang disisipkan namun tidak menambah ukuran berkas *audio* dari *audio* asli, namun metode *low bit coding* tidak tahan terhadap pengujian ataupun pemrosesan *audio* sehinga file *watermark* tidak dapat di ekstrasi kembali.

Tinjauan pustaka selanjutnya yaitu penelitian yang dilakukan oleh Suri Syafitri (2013) skripsinya yang berjudul "Analisis Perbandingan Metode *Low Bit Coding* dan *Least Significant Bit* untuk Digital *Watermarking* pada File WMA". Penelitianya menjelaskan bahwa metode *Low Bit Coding* mampu melakukan peletakan pola *watermark* bedasarkan kehendak penggunanya, dan juga metode ini

lebih aman dari metode *Least Significant Bit* namun memiliki tingkat kerusakan yang lebih tinggi.

Kemudian tinjauan pustaka dari penelitian yang dilakukan oleh Klaveryus Irvan Pakka (2010) dengan skripsinya berjudul "Perbandingan Metode *Phase Coding* dan *Echo Data Hiding* dengan keamanan data menggunakan algoritma *Rijndael* pada steganografi berkas audio". Dalam penelitiannya menyatakan Metode *echo data hiding* lebih baik dalam hal kecepatan proses penyisipan dan kapasitas berkas penampung lebih besar dibanding dengan metode Phase Coding.

Selanjutnya tinjauan pustaka yang digunakan dari penelitian yang dilakukan oleh Arfandy Saputra Edy (2013) dalam skripsinya yang berjudul "Penyisipan Pesan *Text* dengan Metode *Steganografi* pada File Gambar". Dimana perancangan aplikasi steganografi ini menggunakan metode *Least Significant Bit* dalam melakukan penyisipan pesan *text* kedalam file gambar.

Tabel 2.1 Perbandingan Tinjauan Pustaka Antar Peneliti

Peneliti	Tujuan	Metode yang digunakan	Objek yang diteliti	Hasil
Yasin, Verdi Dkk, (2011)	Membangun aplikasi yang digunakan sebagai alat untuk melindungi hak cipta *audio digital.*	*Phase Coding* dengan menggunakan *Fast Fourier Transform (FFT).*	Berkas *Audio* Berformat wave *(.WAV)* Data sisip berupa file *Text (.txt).*	Metode *Phase Coding* menghasilkan kualitas yang baik pada berkas *watermaked audio* karna nilai *psnr* masing - masing *audio* tersebut diatas 30db. Metode Phase coding tahan terhadap serangan *resampling* tetapi tidak tahan terhadap serangan *cropping*, *time streching*, dan *mutiple watermark* dengan metode yang sama.
Soleh, Muhamad (2010)	Mengimplementasi proses penyisipan dan ekstrasi informasi *watermaking* dengan algorithma *AES*. Melakukan Analisis Perbandingan terhadap kualitas dan ukuran data pada file audio sebelum dan sesudah melalui proses penyisipan dan estraksi	*Low Bit Coding* / *Least Significant Bit*. Algorithma Enkripsi yang digunakan *Advanced Ecryption Standart (AES).*	File *Audio* pada berkas *.WAV* Data Sisip Berupa *Text*.	Metode *low bit coding* tidak menambah ukuran berkas suara *WAV.* Berdasarkan uji Analisis ketahanan menunjukan data *text* tidak berhasil di estrak. Berdasarkan hasil uji analisis penyisipan file pesan kedalam file *audio* mempengaruhi kualitas suara yang dihasilkan.
Syafitri, Suri (2013)	Mengetahui kelebihan dan kekurangan metode *Low Bit Coding* dan *Least Significant Bit.*	*Low Bit Coding vs Least Significant Bit*	Berkas audio berformat *(.wma)* Data sisip yang digunakan *text (.txt, .doc, .docx)*	Tingkat kerusakan *audio* yang dihasilkan oleh metode *Low Bit Coding* lebih besar dari metode *Least Significant Bit.* Metode *Low Bit Coding* memiliki keamanan *watermark* yang lebih di bandingkan *Least Significant Bit* dan memiliki pola peletakan yang dapat diatur oleh pengguna-nya.

Klaverius, Irvan (2010)	Membandingkan performa dari kedua teknik steganografi dalam hal kecepatan proses, jumlah informasi yang di tampung *cover* media, dan ketahanan.	*Phase Coding vs Echo Data Hiding* Algorithma Enkripsi *Rijndael*	Berkas Audio berformat *(.Wav)* Data sisip yang digunakan berformat *(.txt)*	Metode *Echo Data Hiding* lebih baik dalam hal kecepatan proses penyisipan dan kapasitas berkas penampung lebih besar dibanding dengan metode *Phase Coding.* Metode *phase coding* lebih baik dalam hal ketahanan kompresi.
Saputra, Arfandy (2013)	Merancang aplikasi Penyisipan Pesan Teks dengan Menggunakan Metode *Steganografi* pada media File Gambar.	*Least Significant Bit (LSB)*	Berkas *Image (.jpeg)* dan output ber-etensi *(.bmp)* Data sisip *text (.txt)*	Aplikasi dapat mendeteksi file gambar yang berisi pesan *text*. Aplikasi dapat melakukan penyisipan tanpa merusak media penampung dan dapat dilakukan *decoding (pengungkapan pesan kembali.*

2.2 Dasar Teori

2.2.1 Audio Watermaking

Audio Watermarking adalah teknik *watermarking* dengan *audio* sebagai berkas yang menjadi tumpangan. *Watermarking* merupakan suatu bentuk dari s*teganography* yaitu ilmu yang mempelajari bagaimana menyembunyikan suatu data pada data yang lain dan mempelajari teknik-teknik bagaimana penyimpanan suatu data (digital) ke dalam data *host* digital yang lain. Istilah *host* digunakan untuk data/sinyal digital yang ditumpangi. *Watermarking* (tanda air) ini agak berbeda dengan tanda air pada uang kertas. Tanda air pada uang kertas masih dapat kelihatan oleh mata telanjang manusia (dalam posisi kertas yang tertentu), tetapi *watermarking* pada media digital disini dimaksudkan tak akan dirasakan kehadirannya oleh manusia tanpa alat bantu mesin pengolah digital seperti komputer, dan sejenisnya. Jadi *watermarking* merupakan suatu cara untuk penyembunyian atau penanaman data/informasi tertentu (baik hanya berupa catatan umum maupun rahasia) ke dalam suatu data digital lainnya, tetapi tidak diketahui kehadirannya oleh indera manusia (indera penglihatan atau indera pendengaran), dan mampu menghadapi proses-proses pengolahan sinyal digital sampai pada tahap tertentu(Nababan, Alexandro 2011:xxxiii).

Aspek - aspek yang harus diperhatikan dalam penyembunyian data diantaranya adalah (Dianty, Indah Wike : 2013):

1. *Fidelity*. Penambahan pesan rahasia ke dalam media penampung tidak mengalami perubahan yang signifikan. Artinya, mutu media penampung

setelah ditambahkan pesan rahasia tidak jauh berbeda dengan mutu media penampung sebelum ditambahkan pesan.

2. *Recovery.* Pesan rahasia yang telah disisipkan dalam media penampung harus dapat diekstrak kembali. Hal ini merupakan syarat mutlak dalam sebuah algoritma watermarking, karena ada banyak cara penyisipan pesan yang tidak terdeteksi namun sulit dalam pembacaan kembali pesan yang telah disisip tersebut.
3. *Robustness.* Pesan yang disembunyikan harus tahan terhadap berbagai operasi manipulasi yang dilakukan pada media penampung. Bila pada media penampung dilakukan operasi manipulasi *audio*, maka pesan yang disembunyikan seharusnya tidak mengalami kerusakan (tetap valid jika diekstraksi kembali).

2.2.2 Metode *Least Significant Bit*

Metode *Least Significant Bit (LSB)* adalah bagian dari barisan data biner (basis dua) yang mempunyai nilai paling tidak berarti atau paling kecil. Letaknya adalah paling kanan dari barisan bit. Sedangkan *most significant bit* adalah sebaliknya, yaitu angka yang paling berarti/paling besar dan letaknya disebelah paling kiri.

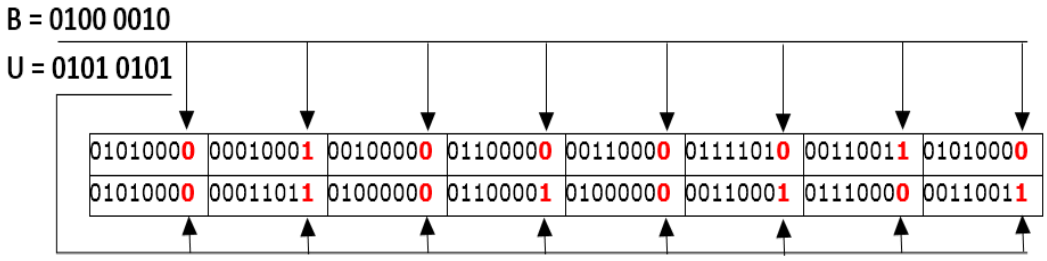
Dasar dari algoritma *least significant bit* ini yaitu dengan cara mengganti bit terakhir dari *sample audio* dengan pesan yang akan disisipkan. Pada saat melakukan penyisipan pesan ke dalam *sample audio*, maka *sample audio* dan pesan penyisip harus diubah menjadi biner terlebih dahulu. Pengubahan *sample audio* menjadi biner dimaksud untuk mendapatkan satu bit pada satu *sample audio*,

dimana bit terakhir dari *sample* biner tersebut akan digantikan dengan satu bit dari bit penyisip.

Sebagai contoh diberikan nilai *sample audio* 50, 10, 40, 60, 30, 7A, 32, 2A, 50, 1B, 40, 60, 40, 30, 70, 32. Pada *sample audio* diatas dilakukan konversi ke biner menjadi larik *sample audio* biner seperti berikut:

01010000, 00010000, 00100000, 01100000, 00110000, 01111010, 00110010, 00101010, 01010000, 00011011, 01000000, 01100000, 01000000, 00110000, 01110000, 00110010.

Pesan penyisip berupa teks "BU" dalam biner adalah 01000010, 01010101. Penyisipan LSB dilakukan dengan mengganti 1 bit dari *sample audio* dengan 1 bit dari bit teks "BU". Dalam bentuk matriks *sample audio* dapat dilihat seperti pada Gambar 2.1



Gambar 2.1 Data dari penyisipan teks "BU"

Pada gambar diatas nilai *sample audio* yang berubah dapat dilihat dimana yang berubah adalah nilai bit yang terakhir (Dianty, Indah Wike : 2013).

Proses Penyisipan LSB

Proses penyisipan (*embedding*) dengan menggunakan metode *least significant bit (LSB)* ini dilakukan penyisipan data atau pesan dengan cara menggantikan bit terakhir dengan bit pesan. Proses penyisipan (*embedding*)

watermark menggunakan metode *least significant bit* adalah sebagai berikut (Syafitri, Suri : 2013).

1. Masukkan file audio format .*wma* yang akan *diwatermark.*
2. Masukkan pesan yang ingin dijadikan sebagai *watermark* (pesan dapat diinput langsung atau berbentuk dokumen).
3. Ubah file audio format .*wma* ke dalam byte.
4. Ubah pesan yang ingin disisipkan menjadi bilangan biner.
5. Ubah pesan ke dalam byte.
6. Ganti nilai bit terakhir *audio* dengan bit pesan yang akan disisipkan.
7. Petakan menjadi file audio .*wma* baru.

Contoh :

Menyisipkan karakter A ke dalam file *audio*. Sebuah pesan huruf A akan disisipkan ke dalam file *audio*.

Penyelesaiannya :

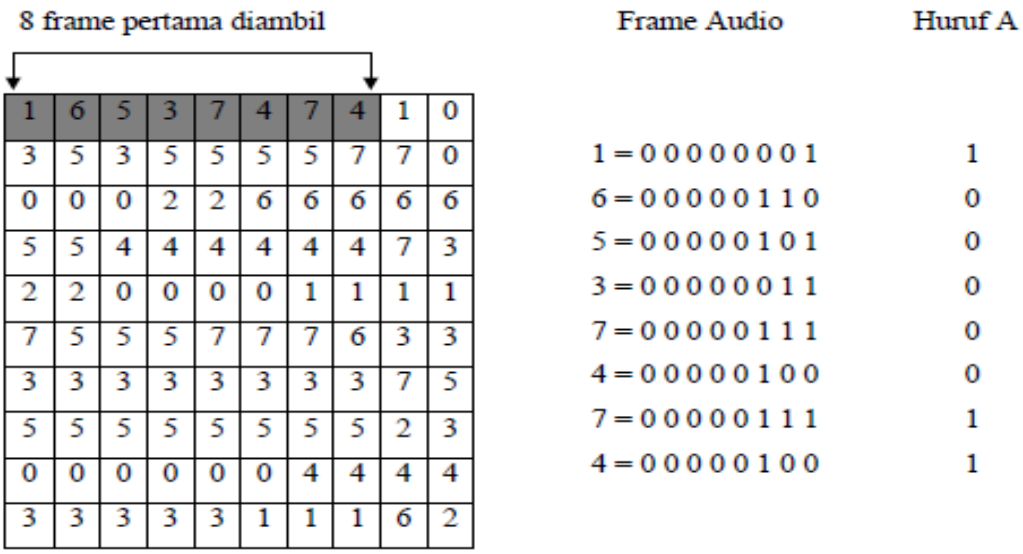
File audio

1	6	5	3	7	4	7	4	1	0
3	5	3	5	5	5	5	7	7	0
0	0	0	2	2	6	6	6	6	6
5	5	4	4	4	4	4	4	7	3
2	2	0	0	0	0	1	1	1	1
7	5	5	5	7	7	7	6	3	3
3	3	3	3	3	3	3	3	7	5
5	5	5	5	5	5	5	5	2	3
0	0	0	0	0	0	4	4	4	4
3	3	3	3	3	1	1	1	6	2

Gambar 2. 2 *Frame audio* (Syafitri, Suri : 2013)

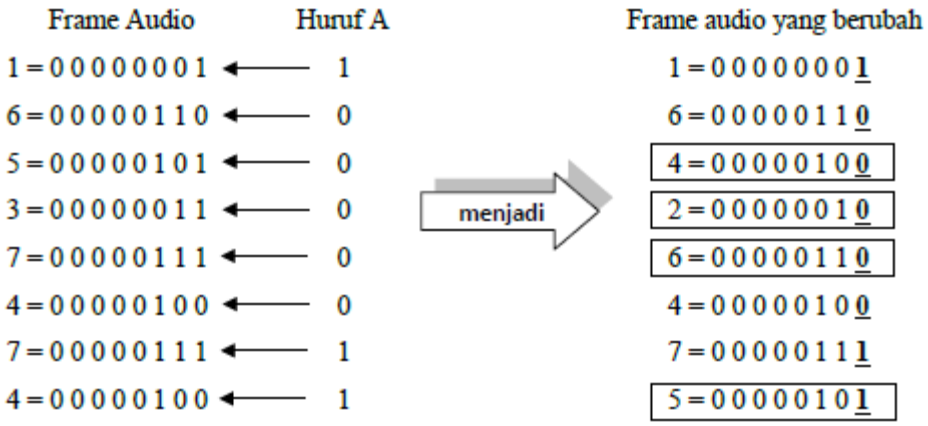
Langkah pertama adalah mengubah kedua data tersebut (huruf A dan file *audio*) menjadi biner.

Nilai biner untuk A adalah 10000011. Karena jumlah digit biner huruf A hanya 8 bit maka jumlah *frame audio* yang dibutuhkan cukup 8 frame saja. Perhatikan 8 frame pertama dari *audio* yang diubah menjadi biner.



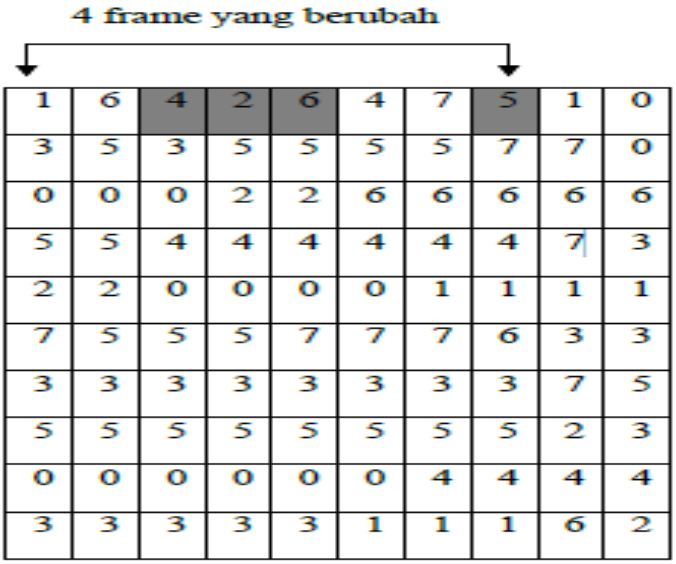
Gambar 2.3 Pengubahan file *audio* menjadi biner

Langkah terakhir adalah mengganti bit terakhir (LSB) dari *frame audio* dengan bit-bit dari huruf A.



Gambar 2. 4 Penyisipan kedalam LSB

Perhatikan bit-bit yang ditandai dengan kotak. Bit-bit *frame audio* mengalami perubahan (dalam hal ini yang berubah hanya 4 frame saja) sehingga audio berubah menjadi :



1	6	4	2	6	4	7	5	1	0
3	5	3	5	5	5	5	7	7	0
0	0	0	2	2	6	6	6	6	6
5	5	4	4	4	4	4	4	7	3
2	2	0	0	0	0	1	1	1	1
7	5	5	5	7	7	7	6	3	3
3	3	3	3	3	3	3	3	7	5
5	5	5	5	5	5	5	5	2	3
0	0	0	0	0	0	4	4	4	4
3	3	3	3	3	1	1	1	6	2

Gambar 2. 5 Perubahan yang terjadi pada *frame audio*

Tampak bahwa *frame-frame audio* yang mengalami perubahan hanya ± 1 intensitas saja. Maka, secara kasat pendengaran hal ini tidak begitu berpengaruh. Selain itu, tidak semua *frame* mengalami perubahan intensitas.

Proses Ekstraksi (*Extraction*)

Proses ekstraksi (*extraction*) *watermark* menggunakan metode *Least Significant Bit* adalah sebagai berikut (Syafitri, Suri : 2013):

1. Masukkan file audio .*wma* yang telah *diwatermark*.
2. Ubah nilai audio .*wma* yang telah *diwatermark* ke dalam byte.
3. Ambil setiap bit terakhir audio .*wma.*
4. Blok setiap 8 bit menjadi {b1, b2, …., bn}.
5. Ubah setiap blok menjadi teks.

Contoh :

Mendeteksi *watermark* yang disisipi pada file audio .*wma*.

1	6	4	2	6	4	7	5	1	0
3	5	3	5	5	5	5	7	7	0
0	0	0	2	2	6	6	6	6	6
5	5	4	4	4	4	4	4	7	3
2	2	0	0	0	0	1	1	1	1
7	5	5	5	7	7	7	6	3	3
3	3	3	3	3	3	3	3	7	5
5	5	5	5	5	5	5	5	2	3
0	0	0	0	0	0	4	4	4	4
3	3	3	3	3	1	1	1	6	2

Gambar 2. 6 File *audio* yang telah *diwatermark*

Langkah pertama adalah mengubah file audio *.wma* menjadi biner

1	6	4	2	6	4	7	5	1	0
3	5	3	5	5	5	5	7	7	0
0	0	0	2	2	6	6	6	6	6
5	5	4	4	4	4	4	4	7	3
2	2	0	0	0	0	1	1	1	1
7	5	5	5	7	7	7	6	3	3
3	3	3	3	3	3	3	3	7	5
5	5	5	5	5	5	5	5	2	3
0	0	0	0	0	0	4	4	4	4
3	3	3	3	3	1	1	1	6	2

Frame Audio

1 = 0 0 0 0 0 0 0 1
6 = 0 0 0 0 0 1 1 0
4 = 0 0 0 0 0 1 0 0
2 = 0 0 0 0 0 0 1 0
6 = 0 0 0 0 0 1 1 0
4 = 0 0 0 0 0 1 0 0
7 = 0 0 0 0 0 1 1 1
5 = 0 0 0 0 0 1 0 1

Gambar 2. 7 Pengubahan file *audio* menjadi biner

Langkah selanjutnya adalah mengambil setiap bit terakhir *audio .wma*.

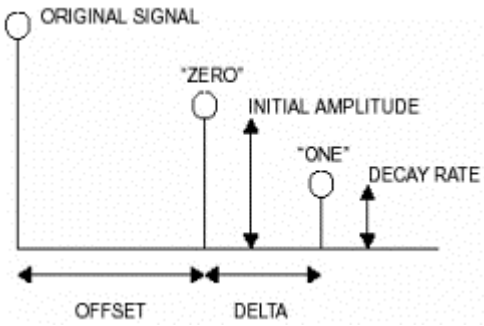
1 = 0 0 0 0 0 0 0 **1**
6 = 0 0 0 0 0 1 1 **0**
4 = 0 0 0 0 0 1 0 **0**
2 = 0 0 0 0 0 0 1 **0**
6 = 0 0 0 0 0 1 1 **0**
4 = 0 0 0 0 0 1 0 **0**
7 = 0 0 0 0 0 1 1 **1**
5 = 0 0 0 0 0 1 0 **1**

Gambar 2.8 Bit terakhir *audio* yang diambil

Langkah selanjutnya adalah menyusun bit menjadi blok yang terdiri 8 bit. Karena pada contoh ini hanya ada 8 bit yang berubah, maka hanya ada 1 blok saja yaitu : **1 0 0 0 0 0 1 1**. Setelah itu blok tersebut kita ubah menjadi *plainteks*. Maka *watermark* yang terbaca yaitu huruf ”A”.

2.2.3 Metode Echo Hiding

Metode *Echo Hiding* melakukan penyisipan data kedalam data suara digital dengan menambahkan *echo* pada sinyal suara. Data yang akan disembunyikan dalam bentuk *echo* dinyatakan dengan variasi dari tiga parameter, yaitu *initial amplitude*, *decay rate*, dan *offset (delay). Initial Amplitude* menyatakan amplitudo asal dari data suara tersebut, *decay rate* menyatakan besar *echo* yang akan diciptakan, dan *offset* menyatakan jarak antara sinyal suara dengan *echo* dalam bentuk fasa sudut dalam persamaan analog. Jika *offset* dari sinyal asal dan *echo* berkurang, maka kedua sinyal akan bercampur. *Echo* ini akan terdengar sebagai resonansi(Piarsa, Nyoman, I Made Ady 2010:191).



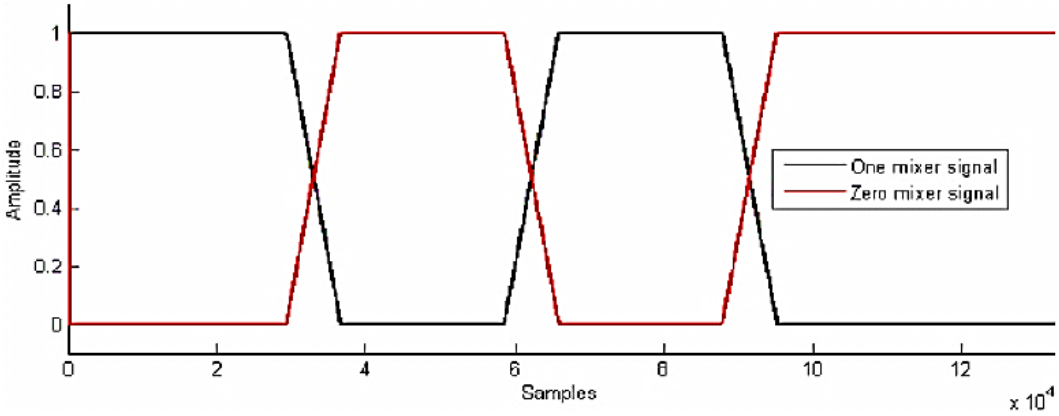
Gambar 2.9 Tiga Parameter dalam metode Echo Hiding

Metode *Echo Hiding* menggunakan dua *delay* waktu, pertama untuk merepresentasikan bit 0 *(offset)* dan yang lainnya untuk menyatakan bit 1 *(offset + delta)*. Kedua *delay* waktu harus berada dibawah *threshold*/batas dimana pendengaran manusia tidak dapat mengenali *echo*. Untuk mengurangi nilai dari

delay waktu agar berada dibawah threshold maka dapat pula dilakukan penyesuaian nilai pada *decay rate* dan *initial amplitude* berdasarkan batas dari sistem pendengaran manusia.

Dalam proses *encode* penyisipan pesan dalam *audio* dengan cara Echo Hiding, sinyal *audio* yang akan disisipi pesan harus dibagi-bagi menjadi beberapa *blok/window*. Setelah itu, dua waktu delay digunakan untuk melakukan proses *encode* data pesan. Misalnya, nilai delay = offset digunakan untuk meng-*encode* biner 0, dan nilai delay = offset + delta digunakan untuk meng-*encode* biner 1. Selain itu, beberapa fungsi serta teknik filter digunakan untuk melakukan proses encode. Persamaan FIR Filter merupakan filter yang umum digunakan untuk melakukan *encode* pesan kedalam file *audio*. Dengan persamaan ini, diperoleh nilai delay dari sinyal *audio*.

Ada dua pulsa yang digunakan di persamaan ini. Satu pulsa untuk menyalin sinyal asli, sedangkan pulsa lainnya digunakan untuk menciptakan *echo* yang tidak mudah dideteksi. Proses *mixer* sinyal untuk encode data dalam bentuk biner dapat dilihat seperti pada Gambar 2.10

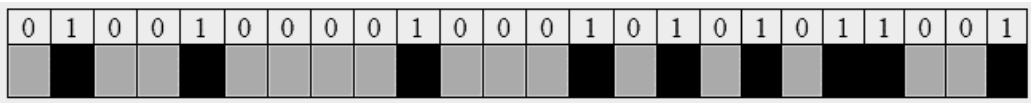


Gambar 2.10 Proses Mixer Signal untuk Encoding Data.

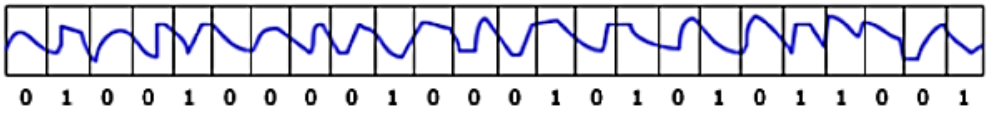
Untuk membentuk *echo* hanya menggunakan dua buah impuls yang disebut *kernel*. Kernel "satu" dibuat dengan delay δ1 detik sedangkan kernel "nol" dibuat dengan delay δ0 detik.

Proses Penyisipan *(Embedding)*

Pertama sinyal dibagi ke dalam blok-blok dan setiap blok diisi dengan 1 atau 0 berdasarkan pesan yang disimpan. Sebagai contoh, pesan yang akan disisipkan ke dalam file audio ialah text "HEY" dengan nilai biner 01001000 01000101 01011001 dan selanjutnya dibentuk menjadi blok sinyal seperti pada Gambar 2.11.

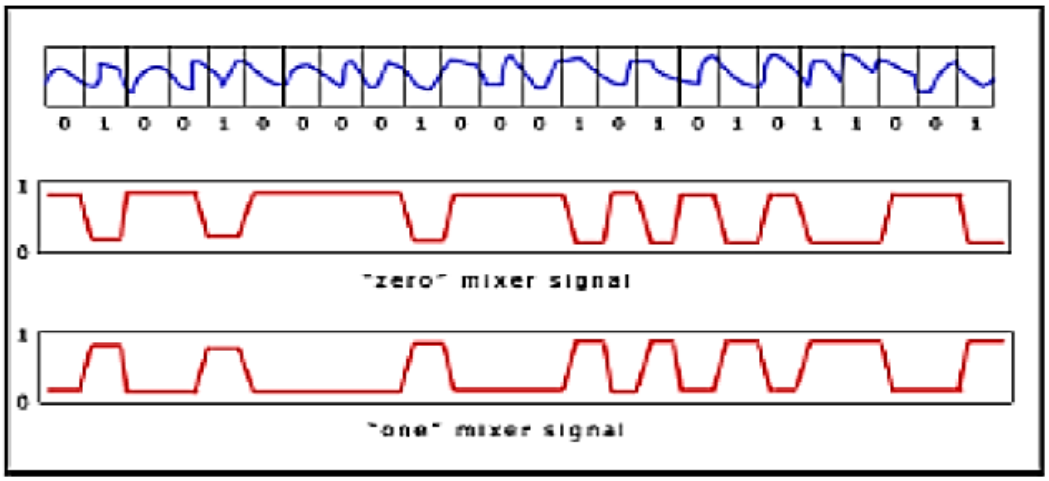


Gambar 2.11 Nilai Biner Sinyal



Gambar 2.12 Contoh Blok Sinyal

Blok-blok tersebut dikombinasikan untuk menghasilkan sinyal baru menjadi seperti pada Gambar 2.12.



Gambar 2.13 dua buah sinyal gabungan

Sinyal *echo* "1" kemudian dikali dengan sinyal *mixer* "1" dan sinyal *echo* "0" dikali dengan sinyal *mixer* "0". Kemudian kedua hasil tersebut dijumlahkan untuk

mendapatkan sinyal akhir. Dengan adanya *offset* dari *echo* dan sinyal asli maka *echo* akan tercampur dengan sinyal aslinya.

2.2.4 Waveform Audio File Format (Wave)

WAV adalah singkatan dari istilah dalam bahasa inggris *Waveform Audio File Format*, merupakan format standar berkas *audio* yang dikembangkan Microsoft dan IBM. Walaupun *WAV* dapat menampung *audio* dalam bentuk terkompresi, umumnya format *WAV* merupakan *audio* yang tidak terkompres. *WAV* menggunakan teknik *pulse-code modulation (PCM)* yang tidak dikompres. Dengan cara ini, detil tidak hilang ketika *audio* analog didigitalkan dan disimpan. Ini membuat format *WAV* (menggunakan PCM) menjadi pilihan untuk mengedit *audio high-fidelity*(Soleh, Muhamad 2010:18).

Berkas WAV sendiri terdiri dari tiga chunk informasi, yaitu RIFF chunk yang mengindentifikasikan bahwa file tersebut adalah file WAV, FORMAT chunk, yang mengidentifikasikan parameter-parameter seperti data rate,dan DATA chunk yang merupakan data yang sebenarnya. Beberapa chunk dapat dipecah sebagai berikut(Klaveryus, Irvan 2010:37):

a. RIFF chunk (12 bytes)

Tabel 2.2 RIFF *Chunk* pada WAVE

Byte Number	
0 – 3	"RIFF" (ASCII Char)
4 – 7	Total Length of Package to Follow
8 – 11	"WAV" (ASCII Char)

b. FORMAT chunk (24 bytes)

Tabel 2.3 FORMAT *Chunk* pada WAV

Byte Number	
0 – 3	"fmt_" (ASCII Char)
4 – 7	Length of FORMAT chunk
8 – 9	Always 0x01 / PCM
10 – 11	channel number
12 -15	Sample Rate
16 – 19	Bytes per second
20 -21	Bytes per sample
22 -23	Bits per sample

c. DATA chunk

Tabel 2.4 DATA Chunk pada WAV

Byte Number	
0 – 3	"data" (ASCII Char)
4 – 7	Length of Data to Follow
8 – end	Data (samples)

2.2.5 Sample Rate

Sample Rate menyatakan banyaknya jumlah *sample* yang dimainkan setiap detiknya. *Sample rate* yang umum dipakai adalah 8000Hz (suara yang dihasilkan menyerupai suara telepon), 11025 Hz (untuk perekaman suara manusia), 22050 Hz (untuk perekaman suara musik) dan 44100 Hz, (sering dipakai dalam *audio* cd karena cocok untuk semua jenis suara) (Soleh, Muhamad 2010:18).

2.2.6 Portable Network Graphic (PNG)

Citra format *PNG* (*Portable Network Graphics*) adalah salah satu format penyimpanan citra yang menggunakan metode pemadatan yang tidak menghilangkan bagian dari citra tersebut (*lossless compression,* yang berarti saat dikompresi kualitas gambar tidak berkurang). Kelebihan file *PNG* adalah adanya warna transparan dan *alpha*. Warna *alpha* memungkinkan sebuah gambar

transparan, tetapi gambar tersebut masih dapat dilihat mata seperti samar - samar atau bening.

2.2.7 Data Format Text *(txt)*

Format data teks merupakan format teks yang digunakan untuk menyimpan huruf, angka, karakter kontrol (tabulasi, pindah baris, dan sebagainya) atau simbol-simbol lain yang biasa digunakan dalam tulisan seperti titik, koma, tanda petik, dan sebagainya. Satu huruf, angka, karakter kontrol atau simbol pada arsip teks memakan tempat satu byte. Berbeda dengan jenis teks terformat yang satu huruf saja dapat memakan tempat beberapa byte untuk menyimpan format dari huruf tersebut seperti font, ukuran, tebal atau tidak dan sebagainya. Kelebihan dari format data teks ini adalah ukuran datanya yang kecil karena tidak adanya fitur untuk memformat tampilan teks. Saat ini perangkat lunak yang paling banyak digunakan untuk memanipulasi format data ini adalah Notepad(Eunike, Johana 2012:21).

2.2.8 SNR (Peak Signal to Noise Ratio)

Salah satu kriteria penyembunyian data rahasia ke dalam media digital mengubah kualitas media adalah Fidelity dimana mutu media penampung tidak jauh berubah. Dimana setelah penambahan data rahasia pengamat tidak mengetahui kalau di dalam media tersebut terdapat data rahasia.

Dalam audio steganografi, pemecah kode menggunakan sistem pendengaran manusia sama baiknya dengan menggunakan analisis komputer dalam mendeteksi device. Sebagai perbandingan pendekatan, spektogram dari sinyal asli dan yang sudah terubah biasanya diberikan atau dihadirkan dalam bentuk data. Spektogram adalah plot sederhana yang berisi frekuensi dari konten sinyal audio

sebagai fungsi waktu. Dalam beberapa kasus tertentu, kualitas dari file audio yang berisi data tersembunyi bisa diperkirakan menggunakan perceptual audio measure, noise to-mask ratio, atau Signal to-Noise Ratio (SNR).

Biasanya, Signal to-Noise Ratio (SNR) dari berkas audio original dibandingkan dengan audio steganografi dapat digunakan sebagai suatu ukuran mutu yang kuantitatif

$$SNR = 10\log_{10}\left\{\frac{\sum_{n=0}^{n-1} x^2(n)}{\sum_{n=0}^{n-1}\left(\tilde{x}(n)-x(n)\right)^2}\right\} \quad (2)$$

Dimana, Xn = sinyal berkas original, $\tilde{X}$ = sinyal berkas steganografi dan n adalah jumlah sample. Semakin tinggi SNR, maka semakin jelas juga ketersembunyian pesan yang ada. Meski jumlah yang dapat ditolerir dari noise bergantung pada aplikasi steganografi dan karakteristik-karakteristik dari unstegano, akan tetapi distorsi noise yang jelas memiliki nilai SNR = 35dB (Sujana, Dadang 2010:34).